COMPUTER SCIENCE

A. K. Watson Hall, 203.432.1246
http://cpsc.yale.edu
M.S., M.Phil., Ph.D.

Chair
Zhong Shao

Director of Graduate Studies
Vladimir Rokhlin (108 AKW, 203.432.1283, vladimir.rokhlin@yale.edu)

Professors
Dana Angluin, James Aspnes, Dirk Bergemann,* Ronald Coifman,* Julie Dorsey, Stanley Eisenstat, Joan Feigenbaum, Michael Fischer, David Gelernter, Mark Gerstein,* John Lafferty,* Rajit Manohar,* Drew McDermott (Emeritus), Dragomir Radev, Vladimir Rokhlin,† Holly Rushmeier, Brian Scassellati, Martin Schultz (Emeritus), Zhong Shao, Avi Silberschatz, Daniel Spielman, Leandros Tassiulas,* Nisheeth Vishnoi, Y. Richard Yang, Steven Zucker†

Associate Professors
Mahesh Balakrishnan, Abhishek Bhattacharjee, Sahand Negahban*

Assistant Professors
Yang Cai, Wenjun Hu,* Julian Jara-Ettinger,* Amin Karbasi,* Smita Krishnaswamy,* Ruzica Piskac, Mariana Raykova, Jakub Szefer,* Marynel Vázquez

Senior Lecturer
Stephen Slade

Lecturers
Benedict Brown, James Glenn, Kyle Jensen,* Scott Petersen, Brad Rosen, Andrew Sherman, Xiyin Tang [Sp]

* A secondary appointment with primary affiliation in another department or school.
† A joint appointment with another department.

FIELDS OF STUDY
Algorithms and computational complexity, artificial intelligence, data networking, databases, graphics, machine learning, programming languages, robotics, scientific computing, security and privacy, and systems.

RESEARCH FACILITIES
The department operates a high-bandwidth, local-area computer network based mainly on distributed workstations and servers, with connections to worldwide networks. Workstations include Dell dual-processor PCs (running Linux or Windows/XP). Laboratory contains specialized equipment for graphics, vision, and robotics research. Various printers, including color printers, as well as image scanners, are also available. The primary educational facility consists of thirty-seven PC workstations supported by a large Intel PC server. This facility is used for courses and unsponsored research by Computer Science majors and first-year graduate students. Access to computing, through both the workstations and remote login facilities, is available to everyone in the department.

SPECIAL ADMISSIONS REQUIREMENTS
Applicants for admission should have strong preparation in mathematics, engineering, or science. They should be competent in programming but need no computer science beyond that basic level. The GRE General Test is required.

SPECIAL REQUIREMENTS FOR THE PH.D. DEGREE
There is no foreign language requirement. To be admitted to candidacy, a student must (1) pass ten courses (including CPSC 690 and CPSC 691) with at least two grades of Honors, the remainder at least High Pass, including three advanced courses in an area of specialization; (2) take six advanced courses in areas of general computer science; (3) successfully complete a research project in CPSC 690, CPSC 691, and submit a written report on it to the faculty; (4) pass a qualifying examination in an area of specialization; (5) be accepted as a thesis student by a regular department faculty member; (6) serve as a teaching assistant for two terms at a TF level 10; and (7) submit a written dissertation prospectus, with a tentative title for the dissertation. To satisfy the distribution requirement (requirement 2 above), the student must take one course in programming languages or systems, one programming-intensive course, two theory courses, and two in application areas. In order to gain teaching experience, all graduate students are required to serve as teaching assistants for two terms during their first three years of study. All requirements for admission to candidacy must be completed prior to the end of the third year. In addition to all other requirements, students must successfully complete CPSC 991, Ethical Conduct of Research, prior to the end of their first year of study. This requirement must be met prior to registering for a second year of study.

MASTER’S DEGREES
M.Phil. See Degree Requirements under Policies and Regulations.

M.S. (en route to the Ph.D.) To qualify for the M.S., the student must pass eight courses at the 500 level or above from an approved list. An average grade of at least High Pass is required, with at least one grade of Honors.
Terminal Master’s Degree Program

Students may also be admitted to a terminal master’s degree program directly. The requirements are the same as for the M.S. en route to the Ph.D. This program is normally completed in one year, but a part-time program may be spread over as many as four years.

A brochure providing additional information about the department, faculty, courses, and facilities is available from the Graduate Coordinator, Department of Computer Science, Yale University, PO Box 208285, New Haven CT 06520-8285; e-mail, cs-admissions@cs.yale.edu.

COURSES

CPSC 521b, Compilers and Interpreters  Staff
Compiler organization and implementation: lexical analysis, formal syntax specification, parsing techniques, execution environment, storage management, code generation and optimization, procedure linkage, and address binding. The effect of language-design decisions on compiler construction.

CPSC 522a, Operating Systems  Zhong Shao
The design and implementation of operating systems. Topics include synchronization, deadlocks, process management, storage management, file systems, security, protection, and networking.

CPSC 523b, Principles of Operating Systems  Abraham Silberschatz
A survey of the underlying principles of modern operating systems. Topics include process management, memory management, storage management, protection and security, distributed systems, and virtual machines. Emphasis on fundamental concepts rather than implementation.

CPSC 524b, Parallel Programming Techniques  Staff
Practical introduction to parallel programming, emphasizing techniques and algorithms suitable for scientific and engineering computations. Aspects of processor and machine architecture. Techniques such as multithreading, message passing, and data parallel computing using graphics processing units. Performance measurement, tuning, and debugging of parallel programs. Parallel file systems and I/O.

CPSC 527a or b, Object-Oriented Programming  Staff
Object-oriented programming as a means to efficient, reliable, modular, reusable code. Use of classes, derivation, templates, name-hiding, exceptions, polymorphic functions, and other features of C++.

CPSC 528b, Language-Based Security  Zhong Shao
Basic design and implementation of language-based approaches for increasing the security and reliability of systems software. Topics include proof-carrying code; certifying compilation; typed assembly languages; runtime checking and monitoring; high-confidence embedded systems and drivers; and language support for verification of safety and liveness properties.

CPSC 531a, Computer Music: Algorithmic and Heuristic Composition  Scott Petersen
Study of the theoretical and practical fundamentals of computer-generated music. Music and sound representations, acoustics and sound synthesis, scales and tuning systems, algorithmic and heuristic composition, and programming languages for computer music. Theoretical concepts are supplemented with pragmatic issues expressed in a high-level programming language.

CPSC 532b, Computer Music: Sound Representation and Synthesis  Scott Petersen
Study of the theoretical and practical fundamentals of computer-generated music, with a focus on low-level sound representation, acoustics and sound synthesis, scales and tuning systems, and programming languages for computer music generation. Theoretical concepts are supplemented with pragmatic issues expressed in a high-level programming language. Prerequisite: ability to read music.

CPSC 534b, Topics in Networked Systems  Yang Yang
Study of networked systems such as the Internet and mobile networks which provide the major infrastructure components of an information-based society. Topics include the design principles, implementation, and practical evaluation of such systems in new settings, including cloud computing, software-defined networking, 5G, Internet of things, and vehicular networking.

CPSC 537a, Introduction to Database Systems  Abraham Silberschatz

CPSC 546a or b, Data and Information Visualization  Holly Rushmeier
Visualization is a powerful tool for understanding data and concepts. This course provides an introduction to the concepts needed to build new visualization systems, rather than to use existing visualization software. Major topics are abstracting visualization tasks, using visual channels, spatial arrangements of data, navigation in visualization systems, using multiple views, and filtering and aggregating data. Case studies to be considered include a wide range of visualization types and applications in humanities, engineering, science, and social science. Prerequisite: CPSC 223.

CPSC 551b, The User Interface  David Gelernter
The user interface (UI) in the context of modern design, where tech has been a strong and consistent influence from the Bauhaus and U.S. industrial design of the 1920s and 1930s through the IBM-Eames design project of the 1950s to 1970s. The UI in the context of the
windows-menus-mouse desktop, as developed by Alan Kay and Xerox in the 1970s and refined by Apple in the early 1980s. Students develop a detailed design and simple implementation for a UI.

**CPSC 554a, Software Analysis and Verification** Razica Piskac
Introduction to concepts, tools, and techniques used in the formal verification of software. State-of-the-art tools used for program verification; detailed insights into algorithms and paradigms on which those tools are based, including model checking, abstract interpretation, decision procedures, and SMT solvers.

**CPSC 555a, Economics and Computation** Yang Cai
A mathematically rigorous investigation of the interplay of economic theory and computer science, with an emphasis on the relationship of incentive-compatibility and algorithmic efficiency. Particular attention to the formulation and solution of mechanism-design problems that are relevant to data networking and Internet-based commerce.

**CPSC 556b / ENAS 951b, Wireless Technologies and the Internet of Things** Wenjun Hu
Fundamental theory of wireless communications and its application explored against the backdrop of everyday wireless technologies such as WiFi and cellular networks. Channel fading, MIMO communication, space-time coding, opportunistic communication, OFDM and CDMA, and the evolution and improvement of technologies over time. Emphasis on the interplay between concepts and their implementation in real systems. The labs and homework assignments require Linux and MATLAB skills and simple statistical and matrix analysis (using built-in MATLAB functions).

**CPSC 560a, Automata Theory and Formal Languages** Andrew Bridy
Introduction to the theory of automata and formal languages, one of the building blocks of theoretical computer science. Major topics covered are finite automata, pushdown automata, and Turing machines, and their associated languages.

**CPSC 568b, Computational Complexity** James Aspnes
Introduction to the theory of computational complexity. Basic complexity classes, including polynomial time, nondeterministic polynomial time, probabilistic polynomial time, polynomial space, logarithmic space, and nondeterministic logarithmic space. The roles of reductions, completeness, randomness, and interaction in the formal study of computation.

**CPSC 569a, Randomized Algorithms** James Aspnes
Beginning with an introduction to tools from probability theory including some inequalities like Chernoff bounds, the course covers randomized algorithms from several areas: graph algorithms, algorithms in algebra, approximate counting, probabilistically checkable proofs, and matrix algorithms.

**CPSC 572a, Intelligent Robotics** Brian Scassellati
Introduction to the construction of intelligent, autonomous systems. Sensory-motor coordination and task-based perception. Implementation techniques for behavior selection and arbitration, including behavior-based design, evolutionary design, dynamical systems, and hybrid deliberative-reactive systems. Situated learning and adaptive behavior.

**CPSC 573b, Intelligent Robotics Laboratory** Brian Scassellati
Students work in small teams to construct novel research projects using one of a variety of robot architectures. Project topics may include human-robot interaction, adaptive intelligent behavior, active perception, humanoid robotics, and socially assistive robotics.

**CPSC 574a, Computational Intelligence for Games** James Glenn

**CPSC 575a / ENAS 575a, Computational Vision and Biological Perception** Steven Zucker
An overview of computational vision with a biological emphasis. Suitable as an introduction to biological perception for computer science and engineering students, as well as an introduction to computational vision for mathematics, psychology, and physiology students.

**CPSC 577b, Natural Language Processing** Dragomir Radev
Linguistic, mathematical, and computational fundamentals of natural language processing (NLP). Topics include part of speech tagging, Hidden Markov models, syntax and parsing, lexical semantics, compositional semantics, machine translation, text classification, discourse, and dialogue processing. Additional topics such as sentiment analysis, text generation, and deep learning for NLP.

**CPSC 579b, Advanced Topics in Computer Graphics** Julie Dorsey
An in-depth study of advanced algorithms and systems for rendering, modeling, and animation in computer graphics. Topics vary and may include reflectance modeling, global illumination, subdivision surfaces, NURBS, physically based fluids systems, and character animation.

**CPSC 610a, Topics in Computer Science and Law** Joan Feigenbaum
This course focuses on socio-technical problems in computing, i.e., problems that cannot be solved through technological progress alone but rather require legal, political, or cultural progress as well. Examples include but are not limited to computer security, intellectual property protection, cyber crime, cyber war, surveillance, and online privacy. The course is addressed to graduate students in Computer Science who are interested in socio-technical issues but whose undergraduate work may not have addressed them; it is designed to bring these students rapidly to the point at which they can do research on socio-technical problems. Students do term projects (either papers or software artifacts) and present them at the end of the term. In order to ensure that there is enough time for both midterm feedback on project proposals and in-class presentation of the finished projects, enrollment is limited to fifteen. If fewer than fifteen Computer Science graduate students enroll, Yale College undergraduates will be allowed to enroll with permission of the instructor. Prerequisites: the basics
of cryptography and computer security (as covered in CPSC 467), networks (as covered in CPSC 433), and databases (as covered in CPSC 437), or permission of the instructor.

**CPSC 634a, Building an Internet Router**  Staff
This course combines seminar-style readings and discussions with practical, hands-on development of a term-long project. Students read a selection of papers to get both a historical perspective and exposure to current research in networking. Students write reviews of the papers to make sure everyone keeps up with the readings and to develop their (technical) communication skills. Throughout the term, students work in teams to develop a fully functional IP router. Students design the control plane in Python on a Linux host and design the data plane in the new P4 language. Teams must demonstrate that their routers can interoperate with those of the other teams by building a small topology utilizing everyone's router. At the end of the course, teams participate in an open-ended design challenge. Prerequisite: undergraduate networking.

**CPSC 640b, Topics in Numerical Computation**  Vladimir Rokhlin
This course discusses several areas of numerical computing that often cause difficulties to non-numericists, from the ever-present issue of condition numbers and ill-posedness to the algorithms of numerical linear algebra to the reliability of numerical software. The course also provides a brief introduction to “fast” algorithms and their interactions with modern hardware environments. The course is addressed to Computer Science graduate students who do not necessarily specialize in numerical computation; it assumes the understanding of calculus and linear algebra and familiarity with (or willingness to learn) either C or FORTRAN. Its purpose is to prepare students for using elementary numerical techniques when and if the need arises.

**CPSC 659a, Building Interactive Machines**  Marynel Vazquez
This course brings together methods from machine learning, computer vision, robotics, and human-computer interaction to enable interactive machines to perceive and act in dynamic environments. Part of the course examines approaches for perception with a variety of devices and algorithms; the other part focuses on methods for decision-making. The course is a combination of lectures, reviews of state-of-the-art papers, discussions, coding homework, and a final team project. Prerequisites: a basic understanding of probability, calculus, and algorithms is expected, as well as proficiency in Python and high-level familiarity with C++. Students who do not fit this profile may be allowed to enroll with permission of the instructor.

**CPSC 662a or b / AMTH 561a or b, Spectral Graph Theory**  Daniel Spielman
An applied approach to spectral graph theory. The combinatorial meaning of the eigenvalues and eigenvectors of matrices associated with graphs. Applications to optimization, numerical linear algebra, error-correcting codes, computational biology, and the discovery of graph structure.

**CPSC 663b / AMTH 663b, Deep Learning Theory and Applications**  Smita Krishnaswamy
Deep neural networks have gained immense popularity in the past decade due to their outstanding success in many important machine-learning tasks such as image recognition, speech recognition, and natural language processing. This course provides a principled and hands-on approach to deep learning with neural networks. Students master the principles and practices underlying neural networks, including modern methods of deep learning, and apply deep learning methods to real-world problems including image recognition, natural language processing, and biomedical applications. Course work includes homework and a final project—either group or individual, depending on the total number enrolled—with both a written and oral (i.e., presentation) component.

**CPSC 690a or b, Independent Project I**  Staff
By arrangement with faculty.

**CPSC 691a or b, Independent Project II**  Staff
By arrangement with faculty.

**CPSC 692a or b, Independent Project**  Staff
Individual research for students in the M.S. program. Requires a faculty supervisor and the permission of the director of graduate studies.

**CPSC 752b / CB&B 752b / MB&B 752b / MCDB 752b, Biomedical Data Science: Mining and Modeling**  Mark Gerstein and Matthew Simon
Biomedical data science encompasses the analysis of gene sequences, macromolecular structures, and functional genomics data on a large scale. It represents a major practical application for modern techniques in data mining and simulation. Specific topics to be covered include sequence alignment, large-scale processing, next-generation sequencing data, comparative genomics, phylogenetics, biological database design, geometric analysis of protein structure, molecular-dynamics simulation, biological networks, normalization of microarray data, mining of functional genomics data sets, and machine-learning approaches to data integration. Prerequisites: biochemistry and calculus, or permission of the instructor.

**CPSC 800b, Directed Readings**  Staff
By arrangement with faculty.

**CPSC 990a, Ethical Conduct of Research for Master's Students**  Holly Rushmeier
This course meets on four consecutive Fridays.

**CPSC 991a / MATH 991a, Ethical Conduct of Research**  Staff
This course is cross-listed with Math 991a.

0 Course cr